



ANSIBLE

Simplest way to automate apps and IT infrastructure

# Co to jest?

- Ansible to narzędzie do wdrażania oprogramowania i zarządzania konfiguracją typu open source, które umożliwia użycie techniki infrastruktura-jako-kod (*Infrastructure as Code - IaC*).
- Zamiast ręcznie przygotowywać system do działania aplikacji, tworzysz skrypt, który robi to automatycznie.

# Demo

Repozytorium: <https://github.com/Konradbor/ansible-azure>

```
git clone https://github.com/Konradbor/ansible-azure.git  
cd ansible-azure  
ansible-playbook 04-create-vm-with-nginx.yaml
```

# Imperatywny (👉°ワ°)👉👉(°ワ°👉) Deklaratywny

W systemach zarządzających można wyróżnić dwa podejścia:

- Imperatywne - zrób to, potem to, jak się nie powiedzie, to zrób coś innego. Np. "zainstaluj pakiet `nginx`, a potem skopiuj plik `nginx.conf`".
- Deklaratywne - nie interesuje mnie jak to zrobisz, chcę mieć dany efekt. Np. "ta maszyna ma mieć pakiet `nginx` i plik `nginx.conf`".

Przykłady deklaratywnych systemów: Puppet i DSC.

# To jaki jest Ansible?

W zasadzie jest mieszanką tych dwóch typów.

Podejście deklaratywne:

```
- name: Install nginx
  yum:
    name: nginx
    state: present
```

Podejście imperatywne:

```
- name: Copy app
  synchronize:
    src: .net-app/
    dest: /app
```

Zaletą podejścia imperatywnego jest prostota tworzenia konfiguracji.

# Dwa tryby - Ad-hoc

Polecenia ad-hoc sprawdzają się w przypadku zadań, które są rzadko powtarzane. Na przykład, jeśli chcesz wyłączyć wszystkie maszyny na święta, możesz wykonać polecenie ad-hoc bez pisania playbooka.

```
$ ansible atlanta -a "/sbin/reboot" -f 10  
$ ansible atlanta -m ansible.builtin.copy -a "src=/etc/hosts dest=/tmp/hosts"
```

# Dwa tryby - Playbook

Playbook jest uruchamiany w kolejności od góry do dołu.

W każdym playbooku są zdefiniowane maszyny (lub grupy maszyn) na których zostanie uruchomiony.

W nim znajdują się zadania, wykonywane od góry do dołu. Playbooki można łączyć w moduły.

```
- name: update web servers
hosts: webserver
remote_user: root

tasks:
- name: ensure apache is at the latest version
  yum:
    name: httpd
    state: latest
- name: write the apache config file
  template:
    src: /srv/httpd.j2
    dest: /etc/httpd.conf
```

# Główne założenie:

Przed wykonaniem każdego polecenia sprawdzany jest stan. Jeżeli nie różni się od oczekiwanego, to akcja nie jest wykonywana.

```
TASK [Install .NET SDK] *****  
ok: [51.145.134.117]
```

```
TASK [Copy app] *****  
changed: [51.145.134.117]
```

```
TASK [Publish app] *****  
changed: [51.145.134.117]
```

```
TASK [Copy nginx config] *****  
ok: [51.145.134.117]
```



# Uruchamianie playbooka

Normalne uruchomienie

```
$ ansible-playbook nazwa_playbooka.yaml
```

Wyświetlanie  
informacji

dodatkowych

```
$ ansible-playbook nazwa_playbooka.yaml -vvv
```

Nadpisanie zmiennych

```
$ ansible-playbook nazwa_playbooka.yaml  
--extra-vars "name=test location=westeurope"
```


Uruchamianie każdego zadania  
krok po kroku

```
$ ansible-playbook nazwa_playbooka.yaml --step
```

# Windows

- Can Ansible run on Windows?

No, Ansible can only manage Windows hosts. Ansible cannot run on a Windows host natively, though it can run under the Windows Subsystem for Linux (WSL).

“  The Windows Subsystem for Linux is not supported by Ansible and should not be used for production systems. ”

# Windows

Ale za to można zarządzać maszynami z Windowsem.

```
- name: Install all critical and security updates
```

```
  win_updates:
```

```
    category_names:
```

```
      - CriticalUpdates
```

```
      - SecurityUpdates
```

```
    state: installed
```

```
  register: update_result
```

```
- name: Reboot host if required
```

```
  win_reboot:
```

```
  when: update_result.reboot_required
```